

as being unpatentable over an article by Hansen (the Hansen article) in view of Beck and as being unpatentable over U.S. Patent No. 6,006,328 to Drake in view of U.S. Patent No. 5,787,247 to Norin et al.

II. Rejections Based on the Thorne-Beck Combination

A. U.S. Patent No. 5,958,005 (Thorne et al.)

Applicants respectfully remind the Examiner that Thorne is not prior art with respect to claims 1-10, 13-15 and 17 because the filing date for Thorne is July 17, 1997, which is one full month after June 17, 1997, the filing date of U.S. Provisional Application No. 60/049,853, from which this application claims priority. Thus, Thorne cannot be cited as a reference against claims 1-10, 13-15 and 17, the subject matter of which were disclosed in the provisional application.

Thorne purports to disclose a method and system for providing selectable degrees of security for e-mail messages. The system creates in the originating computer an e-mail message having a header that specifies one or more security parameters that control the processing of the e-mail in the recipient computer, such as instructions for irretrievably erasing the e-mail following its storage in the recipient computer and instructions as to whether or not copying, archiving, forwarding and printing of the e-mail is permitted, and the recipient computer processes the e-mail in accordance with the instructions.

Thorne is an example of a typical prior art e-mail system. However, at column 6, line 47, Thorne explains how an e-mail message is created with headers under its system:

The E-Mail application will present a user who enters into an originating computer a "Compose E-Mail command", i.e., a template for specifying the security parameters which the message originator desires.

At col. 7, ln. 43, Thorne discusses recording and purging of e-mails in the system by the "routine maintenance cycle":

Upon the recording of an E-Mail document by any processor in the network which is *using an E-Mail application pursuant to the invention, the processor recognizes the security designation*. Each security designation is

assigned a maximum life for the message or document to which it is applied. Upon recognizing the security designation the recording processor derives from a stored table the life to be assigned to that particular document and calculates the date by which that document should be purged. Upon the arrival of that date the *processor schedules the disk on which the message was recorded for a complete purging of that file* and all deleted files on that disk at the preselected low traffic time of day at which disk purges are to be conducted.

It is apparent from the above passage that the “complete purging,” or deletion, is done by the “E-Mail application pursuant to the invention.” Since Thorne is structured like a typical e-mail system, the message is deleted by remote executable files residing with the remote application files in remote directories.

Whenever data is transmitted across a network, it is done according to the protocol of the particular network on which it is traveling. Different networks have different protocols, but all require sending information in packets, pre-determined information packets in a particular size with a header. The header of each packet, among other functions, identifies the information in its packet and indicates where that information belongs in the corresponding file at the receiving end. In this respect, Thorne uses the prior art approach to data transmission. In col. 8, at ln. 28:

The E-Mail application in use according to the invention responds to the setting of the fields of the template by causing the packet assembler to insert into the message packet header flags to cause each recipient computer or processor to respond to the commands created by completion of the template.

The “packet assembler” is the software tool that creates the packets from the data to be transmitted. Thorne uses the general prior art structure, transmitting data in packets, and merely adds information into a prior art component, namely, the header of a data packet. Thorne has not changed the structure of the e-mail; e-mails are sent as before, in packets with data headers.

The deletion mechanism is further described in Thorne at col. 9, starting at ln. 44:

Referring to FIG. 5 the E-Mail message retrieval procedure is now described. The procedure starts at 510. At 512 the E-Mail files are opened and typically a graphical user interface is displayed. At 514 *the application automatically scans all E-Mail messages* to locate any secure messages which are subject to a ripe purge demand but somehow remain in the client computer. Such

messages which are located are thereupon purged and a notification of such purge is sent to the user as well as to the originator or sender of the message.

At 516 the system ascertains whether the user is requesting to open any E-Mail files. The system simply loops until a request is received. At 518 *the system retrieves designated E-Mail messages and inspects the message header flags*. At 520 this inspection determines whether a secure E-Mail flag has been set. Although not specifically illustrated the determination that the document is secure initiates a purge routine in the particular processor which is reading the header. This routine mandates that the disk will be purged of the message pursuant to the appropriate routine which has been identified. In store and forwarding processors this may be immediately subsequent to receipt of an acknowledgment of the forwarding. ...

It is clear from this passage that the remote application executables do all the work. Put another way, it is the remote application executables that “ascertains whether the user is requesting to open any E-Mail files” and “inspects the message header flags ... determines whether a secure E-Mail flag has been set ... [and] initiates a purge routine....”

Thus, the structure of Thorne is based on the typical e-mail architecture with no new components. Flags are placed in headers that were structured in the same way as the prior art. Functions are carried out in the normal manner as is customary with computers, by remote application executables residing on a server, not by executables that are attached to e-mails. A file that is remote from an e-mail cannot be next to (attached) to an e-mail.

B. U.S. Patent No. 5,903,723 (Beck et al.)

Beck purports to disclose an e-mail system that sends attachment references to a network address rather than sending the attachment itself. The attachment to an e-mail under Beck is not an executable and does not delete any data. Beck does not teach e-mail with attachment document and automatically delete feature.

Beck states at column 7, line 5, “an attachment may in alternative preferred embodiments be automatically deleted after ... a given time limit.” Beck only states the same thing as Thorne, that deleting data by some deletion mechanism that is part of a remote application, not by an

attached executable. This is all that is said on the subject of deletion, and Beck does not go into any detail at all on how to do so.

C. Rejection of Claims 1-10, 13-15 and 17-19

The Examiner states that Thorne discloses a method for creating a self destructing document, comprising the steps of creating an executable module that instructs a computer to automatically delete the document to which the executable module is attached when the document, based on a preselected expiration date is expired; and attaching the executable module to the document. The Examiner states that Thorne also teaches the Private message with the security features such as automatically deleted document after being accessed by the recipient or after giving a time limit or other predetermined events (print, forward, copy, store), notified and user given a warning and option when attempt to process the message as a design choice; that Thorne fails to explicitly teach the executable code is attached to the email (or document); that Beck discloses a e-mail message with attachment, and encryption and decryption keys, automatically deleted by a time limit; and that it would have been obvious to modify Beck's email with attachment with Thorne's system to improve security and reliability on the data processing network.

Applicants respectfully request that the Examiner withdraw his rejection with respect to claims 1-10, 13-15 and 17 based in any way upon Thorne. As stated previously, this application claims priority from U.S. Provisional Application No. 60/049,853, which was filed on June 17, 1997. The application for the Thorne patent was filed on July 17, 1997, one full month later. Therefore, with respect to claims 1-10, 13-15 and 17, the subject matter of which were disclosed in the parent provisional application, Thorne is not prior art and cannot be cited as a reference.

In the response dated August 20, 2001, Applicants argued that Thorne teaches how the e-mail system responds to the "setting of fields" by inserting the data field values into the header of an e-mail via the packet assembler, and the values or "header flags" are read by the recipient computer to delete the e-mail in accordance with the header flags. However, in Thorne, there is

no attached executable module and no self-destruct of the e-mail. Instead, destruction of the data in an e-mail is accomplished by the e-mail system, not by the e-mail or any part thereof.

Applicants further argued that Beck does not bolster the weakness of Thorne, since, in Beck, instead of sending the attachment itself, a reference to the network address of the attachment is attached to the e-mail, and the recipient accesses the attachment by clicking on its reference, causing the recipient computer to retrieve the attachment from the network address. Applicants argued that Beck does not teach, as the Examiner states, “a Email message with attachment... automatically deleted by a time limit”, because the attachment Beck refers to is not the e-mail attachment itself but rather a reference to the network address of the e-mail attachment. Beck merely provides a way to efficiently use restricted bandwidth and storage capacities by not sending entire files as attachments with an e-mail. The e-mail itself remains untouched, even after the expiration of the time limit.

Applicants also argued that, because Thorne deals with e-mail security and Beck deals with efficiently utilizing limited bandwidth and storage capacity, there is no motivation to combine the references. Moreover, the combination of Thorne and Beck results in an e-mail system that attaches network references to e-mails and stores security parameters in the e-mail's header, not a method for creating a self-destructing document or an e-mail messaging system with an attached executable for automatically deleting an expired e-mail.

With respect to claims 2-4, the Examiner states that “Thorne-Beck discloses the executable module as an executable code, program, macro as program software including Email application.” Applicants argued that neither Thorne nor Beck discloses or suggests an executable module, so they cannot, alone or in combination, disclose “the executable module as an executable code, program, macro” Thorne refers to the different physical devices on which software may reside, not to the form that the software program or a set of computer instructions, may take, namely an executable code, executable program or macro as recited in claims 2-4. Applicants also argued that Beck teaches an Internet address attached to an e-mail and does not disclose or suggest an executable code, executable program or macro as recited in claims 2-4.

With respect to claim 5, Applicants argued that Thorne-Beck cannot disclose the step of executing the executable module when the document is opened as a design choice of program software, since neither Thorne nor Beck first discloses or suggests an executable module.

The Examiner rejected claims 6-10, 13-15 and 17-47 and stated that they contain the same limitations that were addressed in rejecting claims 1-5. Applicants disagreed and argued that dependent Claim 10 recites the further condition that "the executable module is configured to overwrite the message with null characters" that is not found in claims 1-5, and that dependent Claim 18 recites the further conditions that "the document is an encrypted document, and wherein the executable module is configured to instruct the computer to decrypt the document if the document is not expired, and to delete the document if the document is expired" that are not in claims 1-5. The same is true of claim 19, which is similar to claim 18 except that it pertains to a message instead of a document.

In responding to the Applicants' argument that the Thorne-Beck combination does not teach a self-destructing document, the Examiner stated that:

a recitation of the intended use of the claimed invention must result in a structural difference between the claimed invention and the prior art in order to patentably distinguish the claimed invention from the prior art. If the prior art structure is capable of performing the intended use, then it meets the claim.

The Examiner maintained that "Thorne taught the Email with the instruction (or executable module) for erasure (or self-destructing) of the data message but Thorne is silent on the attachment. Beck taught the Email with attachment document with automatically delete feature." According to the Examiner, it would have been obvious to combine Thorne and Beck to create an e-mail including an attachment document with self-destructing feature.

Applicants strenuously disagree. The Examiner must look at the prior art references as a whole, not only bits and pieces of the references. "It is impermissible within the framework of section 103 to pick and choose from any one reference only so much of it as will support a given position, to the exclusion of other parts necessary to the full appreciation of what such reference

fairly suggests to one of ordinary skill in the art.” *In re Wesslau*, 353 F.2d at 241, 147 U.S.P.Q. at 393. When the references are considered as a whole, their true nature is revealed and the different structures are apparent.

Independent claim 1 recites:

1. A method for creating a self-destructing document, comprising the steps of:
creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based upon a preselected expiration date, is expired;
attaching the executable module to the document.

Claim 1 requires “creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based upon a preselected expiration date, is expired” and “attaching the executable module to the document.” The executable module is not functional without the document to which it is attached. The document may function without the attached executable, but then, a document by itself is not claimed. Rather, it is an express feature of claim 1 that requires that an executable module be created for each self-destructing document. The executable module “instructs a computer to automatically delete the document to which the executable module is attached.” The instructions that make up the executable module, now part of the overall document, direct a processor to delete the document. Since the executable module is a part of the document, then the instructions to delete the document come from the document itself, thereby accomplishing a true “self-destruct”. The prior art combination has no such teaching or suggestion of creating an executable module for each document.

By contrast, in *Thorne*, the instructions that direct the processor to delete an e-mail do not come from the e-mail itself, but rather from remote application files. This is clearly stated in *Thorne* at col. 11, ln. 21 “the application ... deletes and purges secure messages having erase dates of or past the current date.” There is no “self-destruct,” since the e-mails in *Thorne* only contain information for when the e-mail is deleted in its header. Information in a header cannot be considered the same as a file, much less an executable file. There are no instructions in the e-mail itself that direct a processor to perform deletion, and the instructions that delete an e-mail

come from a purging routine conducted by remote system files, part of an e-mail application software package. Therefore, Thorne does not teach “the Email with the instruction (or executable module) for erasure (or self-destructing) of the data message” as maintained by the Examiner, and the “prior art structure” is not “capable of performing the intended use.” Thus, Claim 1 “result[s] in a structural difference between the claimed invention and the prior art”.

Beck is relied upon by the Examiner to teach “the Email with attachment document with automatically delete feature.” Applicants request that the Examiner cite text in Beck to support his assertion. Applicant asserts, contrary to the Examiner’s statement, that documents are not attached to e-mails in Beck. Beck’s e-mail system sends attachment (document) references to a network address, rather than sending the attachment (document) itself. The document, or attachment, stays in the same place (see Abstract, “an attachment is stored in a storage device ... local to the sender”; and col. 7, ln. 19-21, “the attachment file is stored on a WWW HTTP server or otherwise in a memory storage means local to the sending user”). The attachment to an e-mail under Beck is not an executable module and does not delete any data. Thus, neither the e-mail or the attachment in Beck have an “automatically delete feature.”

The only mention of deletion in Beck is at col. 7, lines 5 - 9, which states: “Additionally, ... an attachment may in alternative preferred embodiments be automatically deleted after ... a given time limit.” This is not an automatic delete feature in the e-mail or in the attachment, but rather a delete feature of the e-mail application software, performed by the e-mail application software, the same as in Thorne. Beck only states the same thing as Thorne, deleting data by some deletion mechanism that is part of a remote application, not by an attached executable. This is all that is said on the subject of deletion, and Beck does not go into any detail at all on how to do so. Beck does not teach “the Email with attachment document with automatically delete feature.”

Claims 2-5, 18, and 44-47 depend from and include all the limitations of claim 1. Accordingly, claims 2-5, 18 and 44-47 “result in a structural difference between the claimed invention and the prior art” as well. Applicants point out that the Examiner’s response to

Applicants' arguments still do not point out any grounds for rejection of claim 18 with respect to the unique limitations recited therein, namely encryption and decryption, discussed above.

Independent claim 6 recites:

6. A self-destructing e-mail messaging system, comprising:
 - an executable module, the executable module configured to instruct a computer to automatically delete a message to which the executable module is attached when the message, based upon a preselected expiration date, is expired;
 - an e-mail messaging system, the e-mail messaging system configured to create the message and to transmit the message, the e-mail messaging system attaching the executable module to the message prior to transmission.

Claim 6 recites "an e-mail messaging system comprising...an executable module configured to instruct a computer to delete a message to which the executable module is attached." The executable module, as claimed, is configured to specifically delete a message it is attached to. The deleting executable file in Thorne is not so configured. Thorne uses the same remote executables repeatedly to delete e-mails. Executables are not created for individual e-mails. As a matter of fact, Thorne does not mention the creation of any files other than e-mails, which are not, and cannot, be considered executable modules as they cannot operate on their own without their corresponding e-mail application software.

The system of claim 6 also comprises an "e-mail messaging system configured to create an e-mail message and to transmit the e-mail message, the e-mail message system attaching the executable module to the e-mail message prior to transmission." The Examiner relies on Beck to find this limitation by stating that Beck teaches an "Email with attachment document with automatically delete feature." As already stated above, there is no "attachment document with automatically delete feature" in Beck. Applicants cannot find one, and the Examiner has not pointed one out. E-mail attachments are well known in the art and have been since the inception of e-mail systems. That is all Beck can contribute, e-mail attachments. Even if there were some basis for the argument that Beck had an automatic delete feature, without any teaching of creating an executable module as recited in claims 1 and 6 from Thorne, there cannot be any teaching of "attaching the executable module to the e-mail message prior to transmission" in

Beck. Thus, Claim 6 also “result[s] in a structural difference between the claimed invention and the prior art.”

Claims 7-10, 13-15 and 19 depend from and include all the limitations of claim 6 and are therefore allowable for the same reasons as claim 6. Applicants note that the Examiner’s response to Applicants’ arguments still do not point out any grounds for rejection of claims 10 and 19 with respect to the unique limitations recited therein, as discussed above.

It is clear, from the above, that the “recitation of the intended use of the claimed invention” in claims 1-10, 13-15 and 18-19 does, in fact, “result in a structural difference between the claimed invention and the prior art.” Thorne’s is not capable of creating a self-destructing document, e-mail message, or any type of self-destructing file. Beck does not help because it is structured the same way as Thorne (both references use remote application executable files). By contrast, the claimed invention creates executable modules for each document or e-mail message and attaches the executable module to each document or e-mail.

Independent claim 17 recites:

17. A self-destructing e-mail messaging system, comprising:
- an executable module, the executable module configured to instruct a computer to automatically delete an e-mail message to which the executable module is attached when a predetermined condition is met, wherein said predetermined condition is selected from the group consisting of an attempt to print the message, an attempt to copy the message and an attempt to forward the message;
 - an e-mail messaging system, the e-mail messaging system configured to create the message and to transmit the message, the e-mail messaging system attaching the executable module to the message prior to transmission.

Claim 17 also recites a self-destructing e-mail messaging system with similar limitations to claim 6 except that the executable module of claim 17 is “configured to instruct a computer to automatically delete an e-mail message to which the executable module is attached when a predetermined condition is met, wherein said predetermined condition is selected from the group consisting of an attempt to print the message, an attempt to copy the message and an attempt to

forward the message.” Claim 17, therefore, has the same structural differences from Thorne-Beck as does claim 6 and the cited references thus cannot perform the intended use of claim 17.

D. No *Prima Facie* Case of Obviousness for Claims 1-10, 13-15 and 17-19

In establishing a *prima facie* case of obviousness, the Examiner must show some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art (not based on applicant’s disclosure), to modify the reference or to combine reference teachings; a reasonable expectation of success; and that the prior art reference (or references when combined) teach or suggest all the claim limitations. MPEP 706.02(j)

There is an obvious lack of motivation to combine Thorne with Beck as Thorne deals with e-mail security and Beck deals with efficiently utilizing limited bandwidth and storage capacity. There is no express, implied or inherent suggestion to combine the references. An express suggestion has not been indicated, and the Examiner’s only argument for combining the references is that it is obvious. A combination of references is improper, however, unless the prior art suggests such a combination. *In re Bond*, 910 F.2d 831, 15 USPQ2d 1566 (Fed. Cir. 1990) (the PTO erred in rejecting the claimed invention as an obvious combination of the teachings of two prior art references when the prior art provided no teaching, suggestion or incentive supporting the combination). The Examiner’s mere assertion that it is obvious to combine the references, without finding a suggestion in the prior art, is insufficient to indicate a suggestion or motivation to combine Thorne and Beck, since the suggestion must be express or implied and the Examiner must provide a convincing line of reasoning as to why the combination should be made. The Examiner has not presented any line of reasoning, much less a convincing one to combine Thorne and Beck, other than to say that “By doing so would improve the security and reliability on the data processing network.”

Applicants disagree with the Examiner’s reasoning that combining Thorne and Beck “would improve the security and reliability on the data processing network”. Thorne deals with security, which is evident from its title, “Electronic Mail Security.” Beck deals with bandwidth

(network transmission capacity and speed) and storage capacity. At col. 1, ln. 54, when discussing the problems of the prior art that are addressed by the patent's teachings, Beck states:

There is, therefore, a need for methods and systems for providing for attachments for e-mail messages that more efficiently utilize processor and communications medium bandwidth and memory storage in a computer communications network.

There is no suggestion in Thorne that its system should be modified to more efficiently utilize bandwidth and storage capacity, as taught by Beck. The reverse is also true, there is nothing in Beck to suggest that security is needed to achieve more efficient use of network bandwidth and limited storage capacity. The purported goal of Thorne (enhanced security) has nothing to do with the purported goal of Beck (more efficient network communication).

In addition, combining the two references does not create a synergistic benefit, and neither reference adds anything to the other's objectives. Thorne does not enhance the efficiency of Beck, and Beck does not enhance the security of Thorne, as replacing an attachment with the address of the attachment has no effect on system security. Accordingly, there is not only a lack of motivation, but a total lack of any logical reason to combine Thorne with Beck. Furthermore, neither reference changes the way the other works. Thorne would have no effect on the workings of Beck since Thorne does not deal with attachments at all, and Beck would have no effect on the workings of Thorne since Beck does not manipulate data headers in any way. It seems that the Examiner arrives at the Thorne-Beck combination by his own analysis, which is improper.

In view of the above, it is not readily apparent why combining an e-mail security system with a method to efficiently utilize limited bandwidth and storage capacity is proper. The suggestion to combine the references cannot be taken from the applicant's disclosure, but must be found in the prior art. It is the Examiner's duty to explain why combining Thorne and Beck is proper, and it is impermissible to merely combine the elements of separate references to produce the claimed invention without a suggestion to do so.

The second requirement to establish a *prima facie* case of obviousness is a reasonable expectation of success, i.e., a reasonable expectation that the proposed combination will work as

intended by the applicant's disclosure. This expectation must be found in the prior art, not in the applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991). It follows that there must be a reasonable expectation that Thorne, by placing data fields in headers to delete data at a certain time, when combined with Beck, which attaches network references to e-mails, creates an executable module that automatically deletes a document or e-mail message and attaching such a module to a document or e-mail message to render claims 1 and 6 obvious. In other words, looking at Thorne and Beck, together, without the knowledge of an attached executable module, there should be a reasonable expectation that, as the Examiner sees it, taking the data header from Thorne and the attached network reference from Beck would create an executable module that functions to delete an e-mail or document to which it is attached.

The data header in Thorne is merely a data storing mechanism, it does not function or execute any instructions. Similarly, the network reference of Beck is only a reference with location information, it also does not function or execute any instructions. Neither are executables. It follows that there cannot be a reasonable expectation that a working executable module would result from the combination of Thorne and Beck because neither has any direct disclosure of an executable module, much less an executable module attached to an e-mail. In both references, instructions are executed by and according to a remote executable external to, not attached to, the e-mail as is customary in the prior art. Applicants are not arguing that executable modules do not exist in the prior art, since executable files clearly enable computers to operate. However, it is the executable module that automatically deletes a document or e-mail message to which it is attached, as recited in claims 1 and 6, that does not exist in the prior art.

The third and final requirement for a *prima facie* case of obviousness is the teaching or suggestion of all the claim limitations in the prior art. Applicants' responses to the Examiner's rejections on pages 2-3, paragraphs 1-7, set forth above (pp. 1-10, *infra*), address the many deficiencies in the cited references for reciting all the limitations in claims 1-10, 13-15, and 17-19. The Examiner's response to those arguments at page 7 of the Office Action are set out above and have been successfully refuted (pp. 28-36, *infra*). Neither reference discloses an executable

module attached to a document or e-mail that deletes the document or e-mail when a predetermined condition is met.

E. Rejection of Claims 20-47

The Examiner states that claims 20-47 contain the same limitations as claims 1-5 and are rejected for the same reasons. In the response dated August 20, 2001, Applicants argued that independent Claim 20 teaches a method for creating a virtual container in contiguous memory locations for an encrypted digital object where the virtual container stores an expiration date for the digital object, which is very different from the Claim 1 method of creating a self-destructing document by creating an executable module that automatically deletes a document to which it is attached when the document expires and attaching the executable module to the document.

Applicants argued that the method of creating a virtual container of claim 20 is not the same as the steps of creating an executable module in claim 1. The virtual container resides in contiguous memory locations in a computer with a header portion and a digital object portion, whereby a digital object goes into the digital object portion and information about an expiration date for the digital object goes into the header portion. By contrast, the executable module of claim 1 attaches to a document. Applicants argued that the method for creating a virtual container of Claim 20 contains limitations that are not present in claim 1 and that the Examiner has failed to establish a prima facie case of obviousness and provide adequate grounds for rejection because he has not found each and every limitation of claim 20 taught or suggested by the prior art. MPEP § 2143.3.

Similarly, claim 21 recites a method for extracting a document from a virtual container. Claim 22 recites a virtual container system. Claim 23 recites a method for creating a virtual container and extracting a digital object from a virtual container, and claim 32 recites a method for transmitting a destructible document. Applicants argued that the Examiner has not shown where any of the limitations from any of claims 21-23 and 32, or of their dependent claims, can be found in claims 1-5 or in the prior art and that a prima facie case of obviousness has not been made for these claims.

With respect to the Applicants' arguments regarding virtual container embodiment in claim 20, the Examiner states "the fact that applicant has recognized another advantage which would flow naturally from following the suggestion of the prior art cannot be the basis for patentability when the differences would otherwise be obvious. ... In this case, examiner interprets the virtual container as the executable module of [sic] macro/instructions/digital code or software which is self destruct as taught by Thorne."

The Examiner's position seems to be that the prior art suggested the attached executable embodiment, so the virtual container embodiment is a naturally flowing advantage. Applicants have already explained how the prior art does not suggest the attached executable embodiment of claims 1-10, 13-15, 17-19 and 44-47. Even if the Examiner were correct, his position does not make sense. The two embodiments are completely different structurally. Applicants fail to understand the Examiner's reasoning. Claim 20 is not reciting a different advantage than claim 1, claim 20 recites a different structure.

In accordance with claims 1 and 6, the executable module is attached to an e-mail or document, which is like stapling an envelope to a letter. The virtual container embodiment recited in claims 20-43 contains the e-mail or document, like placing the document in the envelope and sealing the envelope. The two structures are completely different from each other. To consider the two embodiments the same, and provide the same grounds for rejection, implies that claims 1-19 and 20-43 have the same limitations. However, the two sets of claims are very different, and the Examiner has provided no separate and distinct grounds of rejection for the two separate and distinct sets of claims, namely claims 1-19 and claims 20-43.

Among the three independent for the attached executable embodiment, claims 1, 7 and 17, there are some common limitations: an executable module that instructs a computer to automatically delete a document or e-mail message to which it is attached when the document/e-mail message expires or a predetermined condition is met, and attaching the executable module

to the document or e-mail. The independent claims of the virtual container embodiment do not recite either of the common limitations set forth above, as set forth below.

Independent claims 20-23 and 32 recite:

20. A method for creating a virtual container containing a digital object, comprising the steps of :
- creating a virtual container, the virtual container residing in contiguous locations in an electronic storage media of a computer, the virtual container including a header portion and a digital object portion;
 - selecting a digital object for insertion into the virtual container;
 - applying an encryption technique to the digital object to create an encrypted digital object;
 - writing the encrypted digital object into the digital object portion;
 - selecting an expiration date for the digital object;
 - writing information indicative of the expiration date into the header portion of the virtual container.
21. A method for extracting a document from a virtual container, comprising the steps of
- reading information indicative of an expiration date from a header portion of a virtual container, the virtual container residing in contiguous locations in an electronic storage media of a computer, the virtual container including the header portion and a digital object portion, the digital object portion containing an encrypted digital object;
 - determining, based upon said information, if the electronic object is expired;
 - overwriting the digital object portion of the virtual container with null data if the electronic object is expired; and
 - reading the digital object from the digital object portion and applying a decryption technique to the digital object if the digital object is not expired.
22. A virtual container system, comprising:
- a container creator utility, the container creator utility creating a virtual container which resides in contiguous locations in an electronic storage media of a computer, wherein the virtual container includes a header portion and a digital object portion, the container opener utility receiving a digital object selection and an expiration date selection from a user, the container creator applying an encryption technique to the selected digital object to create an encrypted digital object and writing the encrypted digital object into the digital object portion of the virtual container, the container creator writing information indicative of the expiration date into the header portion of the virtual container;
 - a container opener utility, the container opener utility reading the information indicative of the expiration date from the header portion of the virtual container, the container opener determining, based upon said information, if the electronic object is

expired; the container opener overwriting the digital object portion of the virtual container with null data if the electronic object is expired, the container opener reading the encrypted digital object from the digital object portion and applying a decryption technique to the digital object if the digital object is not expired.

23. A method for creating a virtual container and extracting a digital object from a virtual container, wherein the method of creating the virtual container comprises the steps of

- creating a virtual container, the virtual container residing in contiguous locations in an electronic storage media of a computer, the virtual container including a header portion and a digital object portion;

- selecting a digital object for insertion into the virtual container;

- applying an encryption technique to the digital object to create an encrypted digital object;

- writing the encrypted digital object into the digital object portion;

- selecting an expiration date for the digital object; and

- writing information indicative of the expiration date into the header portion of the virtual container;

and wherein the method for extracting the document from the virtual container, comprises the steps of

- reading information indicative of an expiration date from a header portion of a virtual container,

- determining, based upon said information, if the electronic object is expired;

- overwriting the digital object portion of the virtual container with null data if the electronic object is expired; and

- reading the digital object from the digital object portion and applying a decryption technique to the digital object if the digital object is not expired.

32. A method for transmitting a destructible digital object to a recipient, comprising the steps of

- creating a virtual container, the virtual container residing in contiguous locations in an electronic storage media of a computer, the virtual container including a header portion and a digital object portion;

- selecting a digital object for insertion into the virtual container;

- applying an encryption technique to the digital object to create an encrypted digital object;

- writing the encrypted digital object into the digital object portion;

- selecting an expiration date for the digital object;

- writing information indicative of the expiration date into the header portion of the virtual container,

transmitting the virtual container and a container opener utility to a recipient, wherein the container opener utility, when invoked by the recipient, reads the information indicative of the expiration date from the header portion of the virtual container, determines, based upon said information, if the electronic object is expired, overwrites

the digital object portion of the virtual container with null data if the electronic object is expired, and reads the encrypted digital object from the digital object portion and applies a decryption technique to the digital object if the digital object is not expired.

Among the unique limitations in the virtual container embodiment are “contiguous locations,” “a header portion,” and “a digital object portion”. Claims 1-10, 13-15, 17-19 and 43-47 do not mention memory storage and organization. It is not typical to always store and arrange data in contiguous (adjacent) memory locations. Constant deletion of old data and storage of new data creates pockets of available memory space spread out over storage media. Computers use available memory, wherever it may be located, with pointers to reference a particular location where data is stored.

Claim 20 recites “a header portion” and “a digital object portion,” neither of which limitation is recited anywhere in claims 1-10, 13-15 and 17-19. These portions are not the same as an attached executable and a digital document. The “header portion” is a header that contains “information indicative of the expiration date.” There are no instructions contained in the “header portion” like in the attached executable. The digital object portion is for “insertion” of “a digital object,” or for containing the digital object. Referring back to the envelope example, this is similar to inserting a document into an envelope. The “header portion” can be thought of as a label on the envelope indicating the expiration date of the document contained therein. Nothing in the attached executable embodiment contains the document or e-mail message. The executable module attaches to the document or e-mail message. Nothing is inserted into or contained in the executable module of claims 1 and 6 other than its own instructions.

Claim 21 recites a method for extracting a document from a virtual container. Claim 22 recites a virtual container system to implement the method steps in claims 20 and 21; claim 23 recites a method for creating a virtual container and extracting a digital object from a virtual container, combining claims 20 and 21; and in claim 32, a method for transmitting a destructible document. Claims 21-23 and 32 all recite the header portion, digital object portion, and the contiguous locations in an electronic storage media of a computer as in claim 20. Accordingly, they all share the same structural differences from the prior art.

F. No *Prima Facie* Case of Obviousness of Claims 20-47

Applicants have pointed out above the lack of any suggestion to combine Thorne and Beck with respect to claims 1-10, 13-15 and 17-19. The same arguments apply for claims 20-43.

The Examiner has not satisfied the second requirement to establish a *prima facie* case of obviousness, showing a reasonable expectation of success that the proposed combination will work as intended by the applicant's disclosure. There must be a reasonable expectation that Thorne, by placing data fields in headers to delete data at a certain time, when combined with Beck, which attaches network references to e-mails, creates the situation where the virtual container embodiment of claims 20-43 is just "another advantage which would flow naturally from following the suggestion of the prior art ... [and] the differences would otherwise be obvious." Applicants have pointed out the differences between the two embodiments of the present invention such that the virtual container embodiment is not a natural advantage with obvious differences from the attached executable embodiment. Thus, there can be no reasonable expectation of success.

Following the Examiner's reasoning, there should be a reasonable expectation that the combination of Thorne and Beck would produce a method for creating a virtual container containing a digital object, the virtual container residing in contiguous memory locations with a header portion and a digital object portion. At the outset, it is clear that there is no method for creating a virtual container in either Thorne or Beck, alone or in combination. Neither reference even mentions individual memory locations, much less contiguous locations. In fact, the word "container" does not appear in either reference. The concept of containing is not used by either reference expressly, impliedly, or inherently. Without any express, implied, or inherent disclosure of the limitations of claims 20-43 in the Thorne-Beck combination, there can be no reasonable expectation that the present invention could be created from the cited references.

Furthermore, the third and final requirement for a *prima facie* case of obviousness, the teaching or suggestion of all the claim limitations in the prior art, is not met here. There are many deficiencies in the cited references for reciting all the limitations in claims 20-43.

The Examiner states, in response to Applicants' arguments, that he "interprets the virtual container as the executable module." However, the Examiner is impermissibly ignoring the clear difference between the express wording of claims 1- 10, 13-15, 17-19 and 44-47 as compared to claims 20-43. "All words in a claim must be considered in judging the patentability of that claim against the prior art." *In re Wilson*, 424 F.2d 1382, 1385. For example, a "virtual container residing in contiguous locations in an electronic storage media of a computer" is recited in claims 20-23 and 32, all independent claims, and the Examiner has addressed this limitation, which appears in five independent claims. No indication of whether or where the limitation exists in the prior art has been made. The same is true for the limitation "header portion and a digital object portion" recited in those claims. These limitations do not appear anywhere in claims 1-10, 13-15 17-19 and 44-47, and there is nothing in those claims even remotely similar. Accordingly, the Examiner has not found any of the claim limitations in claims 20-43, let alone *all* the claim limitations of claims 20-43.

III. Rejections Based on the Ji-MacPhail Combination

A. U.S. Patent No. 5,889,943 (Ji et al.)

Ji purports to disclose a system for detecting and eliminating viruses by polling and retrieving modules in communication with a postal node of a network e-mail system to determine the presence of unscanned messages and downloading data associated with them for treatment by a virus analysis at the node. The system of Ji does not attach viruses to email, programs or documents and does not use them in any way, but rather instead "*detects*" and "*eliminates*" viruses. The action of "attaching" never occurs in Ji. As stated at column 3, line 55 of Ji:

The present invention also comprises an apparatus for detecting and eliminating viruses which may spread throughout a network in messages accessed with an electronic mail system. In such mail systems, messages directed to a user at a client node are typically stored a postal node prior to their access by the client node. Viruses are detected and corrective action taken by *a mail scanning apparatus which preferably resides at the client node*. The mail scanning apparatus preferably includes: a polling module for determining the presence of unread messages at the postal node, a retrieval module for downloading unread messages to the memory of a client node and a virus analysis and treatment module *for determining whether the message contains a virus and for facilitating corrective action to prevent its spread.* (emphasis added)

Thus, the modules in Ji, parts of the mail scanning apparatus, “reside at the client node” and determine “whether the message contains a virus and ... facilitat[es] corrective action to prevent its spread.”

B. U.S. Patent No. 4,899,299 (MacPhail)

MacPhail relates to a method of managing the retention of electronic documents, including deletion of electronic documents with expiration dates, but the structure disclosed therein adds nothing new to that of Ji. An overview of the MacPhail system structure can be found in column 5, lines 44-61:

FIG. 3a represents an overview of the system from the standpoint of the major interactive steps involved in storing a document that has been created by an End User (EU). Block 50 represents the EU. *Blocks 51 and 52 represent programs stored at the terminal* while block 53 represents local disk storage. *Blocks 54 and 55 represent the library server function located at the host* and is the central depository for stored system documents. The program represented by block 51 is referred to as the Dialogue Manager application and functions to provide the necessary menus and prompts to obtain information from the user. It is essentially the interface to the user from the terminal. The program represented by block 52 is referred to as the *requester application and functions to build requests and send them to the library server*. The function of the library server is to validate/set labels and expiration dates and to file the document. (emphasis added)

The requester application sends requests to delete expired documents to the library server, which is responsible for setting expiration dates and filing documents and which resides on a host computer, not attached to documents being deleted. Deletion is accomplished by the requester application building and sending a request to the library server to delete expired documents. The requester application, block 52, is “stored at the terminal,” and therefore, is not attached as well. Another example of a remote application executable.

C. Rejection of Claims 1-10 and 13-15

Claims 1-10, and 13-15 were rejected under 35 U.S.C. §103 as being unpatentable over U.S. Patent No. 5,889,943 to Ji et al. in view of U.S. Patent No. 4,899,299 to MacPhail. With respect to claim 1, the Examiner stated that Ji discloses a method for creating a self-destructing document, comprising the steps of creating an executable module which instructs a computer to

automatically delete the document to which the executable module is attached (when the document based on a preselected expiration date is expired); and attaching the executable module to the document. The Examiner stated that Ji fails to detail when the document based on a preselected expiration date is expired, but MacPhail discloses a electronic documents is set for automatically delete by an expiration date. Therefore, according to the Examiner, it would have been obvious to incorporate the message automatically deleted based on an expiration date as taught by MacPhail into Ji's system in order to utilize the email message attach by an executable code would automatically delete by an expiration date, since doing so would improve the reliability of data storage on the network.

In the response dated August 20, 2001, Applicants argued that the e-mail virus detection and elimination system of Ji polls postal nodes on a computer network for unscanned messages, downloads any found unscanned messages and performs virus detection and analysis at the node. There is no attached executable module in Ji; rather, the system works by polling postal nodes and deletes viruses, not e-mail. The e-mails are analyzed and treated, not deleted, and are chosen because they are infected with a virus, not because they have expired. All modules discussed in Ji are separate and distinct from the e-mails in the system, and none of the modules attach to any e-mails or delete any e-mails. Ji contains no mention or suggestion of creating an attached executable module to instruct a computer to automatically delete a document or e-mail message, and certainly no disclosure or suggestion to attach an executable module to a document.

Applicants further argued that combining Ji and MacPhail still does not disclose or suggest an executable module attached to a document or e-mail message and configured to instruct a computer to delete the document or e-mail message to which the module is attached when the document or message expires. Both Ji and MacPhail are polling systems that work by polling the contents of a system to find a specific type of data, and such systems are distinguished in the present specification. Combining MacPhail with Ji results in a system that polls its contents for time-expired data, instead of polling for viruses, and deleting the time-expired data.

With respect to the Applicants' arguments against this rejection, the Examiner states that

a prior art structure meets the claims if it is capable of performing the intended use. The Examiner stated that Ji taught an e-mail system with a virus analysis and treatment module wherein the virus (executable code for self-destruction) is attached to e-mail or other program, and MacPhail taught the method of deleting an electronic document with an expiration date, and that it would therefore have been obvious to combine Ji and MacPhail to create a messaging system with self-destruct module attach to an e-mail message as a design choice.

However, contrary to the Examiner's suggestion, the virus in Ji is not executable code for self-destruction and is not attached to the e-mail or other program. Ji does not attach viruses to email, programs, documents, or anything else for that matter, as is clearly stated in the abstract of Ji, which is set forth below, with emphasis added:

The *detection and elimination of viruses* on a computer network is disclosed. An apparatus for detecting and eliminating viruses which may be introduced by messages sent through a postal node of a network electronic mail system includes polling and retrieval modules in communication with the postal node to determine the presence of unscanned messages and to download data associated with them to a node for treatment by a virus analysis and treatment module. A method for detecting and eliminating viruses introduced by an electronic mail system includes *polling* the postal node for unscanned messages, downloading the messages into a memory of a node, and performing virus detection and analysis at the node.

Thus, Ji "*detects*" and "*eliminates*" viruses, but the system does not use them in any way. The action of "attaching" never occurs in Ji.

The concept of attaching a virus (executable code for self-destruction) to an e-mail or other program comes only from Applicants' disclosure, which states:

In accordance with a first embodiment of the present invention, a self-destructing document and e-mail messaging system is provided that automatically destroys documents or e-mail messages at a predetermined time by attaching an executable module such as a "virus" to the document or e-mail message.

Attaching viruses to e-mail as executable modules cannot be found anywhere in the prior art cited by the Examiner. Therefore, the Examiner's conclusion is forbidden, since "The teaching or suggestion to make the claimed combination and the reasonable expectation of success must

both be found in the prior art and not based on applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991)." Also, MPEP 2143.03 states "all claim limitations must be taught or suggested by the prior art."

The modules in Ji, parts of the mail scanning apparatus, "reside at the client node" (col. 3, line 63) and determine "whether the message contains a virus and ... facilitat[es] corrective action to prevent its spread" (col. 3, ln. 67 - col. 4, ln. 2). The module in Ji is not attached to any document or e-mail but rather resides on the client node. The Ji module is a remote application executable like those in Thorne and in the general prior art. In addition, the module deletes viruses, not documents to which the module is attached. Clearly, the structure is completely different than that of the present invention.

The structure of MacPhail is similar to that of Ji and adds nothing new. The Examiner relies on MacPhail only to teach "deletion of an electronic document with an expiration date." The structure of MacPhail has nothing in common with the present invention. In MacPhail, as described at column 5 at lines 44-61, the requester application sends requests to delete expired documents to the library server, which is responsible for setting expiration dates and filing documents. Deletion is accomplished by the requester application building and sending a request to the library server to delete expired documents. The library server resides on a host computer and is not attached to the document being deleted. The requester application is "stored at the terminal" and, therefore, is not attached, and is another example of a remote application executable. Because the structure of MacPhail is not the same as that of the claimed invention, "a structural difference between the claimed invention and the prior art in order to patentably distinguish the claimed invention from the prior art" does exist, contrary to the Examiner's argument.

The method of claim 1 requires "creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based upon a preselected expiration date, is expired" and "attaching the executable module to the document." The executable module is attached to the data being

deleted, becomes part of the document and is not functional without the document to which it is attached. The document may function without the attached executable, but a document by itself is not claimed. There is an executable module attached to each document created in accordance with claim 1. The executable module “instructs a computer to automatically delete the document to which the executable module is attached,” and the instructions that make up the executable module, now part of the overall document, direct a processor to delete the document. Thus, the instructions to delete the document come from the document itself, thereby accomplishing a “self-destruct”. Once the document or e-mail is destroyed, common sense dictates that the executable file cannot be re-used.

In the first sentence of its abstract, Ji states its purpose: “The detection and elimination of viruses on a computer network is disclosed.” Ji never creates an executable module as in claim 1. In fact, nowhere in Ji is there any disclosure of creating any type of file, much less an executable module that instructs a computer to automatically delete the document to which the executable module is attached when the document, based upon a preselected expiration date, is expired. Without the first step of creating an executable module, the remaining steps of claim 1, namely, “attaching the executable module to the document”, also cannot be found in Ji.

Furthermore, like all previously discusses references, Ji fails to perform a self-destruct. The instructions directing the processor to destroy a virus (which is not like the e-mail or document of the present invention) do not come from the virus itself, but from the treatment and analysis module residing on the client node. The treatment and analysis module is not even attached to the virus.

Combining the two references yields a system that scans its contents for viruses and expired data for deletion. Ji purports to disclose a virus detection system that scans the contents of an e-mail system for “detection and elimination of viruses”(Abstract). MacPhail purports to delete expired data by scanning system contents for “managing the retention and deletion of electronic documents” (Abstract). Ji does not change the way MacPhail works, and MacPhail does not change the way Ji works. Ji does not enhance or affect the deletion of time-expired

data. MacPhail does not enhance or affect the virus detection and elimination capabilities of Ji. The resultant system still scans its contents. Even combined, the references still do not disclose attaching a virus to accomplish a self-destruct functionality, as stated by the Examiner.

Claim 6 recites “an e-mail messaging system comprising ... an executable module configured to instruct a computer to delete a message to which the executable module is attached.” The executable module, as claimed, is configured to specifically delete a message it is attached to. There is no such executable file in Ji or MacPhail, as executables are not created for individual e-mails. Instead, Ji-MacPhail uses the same system executable files over and over to delete data residing at a separate location. Clearly, the remote system executable files in Ji and MacPhail are not attached to the data they are deleting. The modules of Ji, discussed earlier, also are not executable module configured to instruct a computer to automatically delete a message to which the executable module is attached when the message, based upon a preselected expiration date, is expired, as recited in claim 6. Likewise, without teaching the executable module of claim 6, the references cannot teach the second element of claim 6, the e-mail messaging system that attaches the executable module prior to transmission.

Claims 7-10, 13-15 depend from and include all the limitations of claim 6 and are therefore allowable for the same reasons as claim 6. Applicants note that the Examiner’s Response to Arguments still do not point out any grounds for rejection of claims 10 with respect to the unique limitations recited therein, discussed above.

The Examiner stated that “a recitation of the intended use of the claimed invention must result in a structural difference between the claimed invention and the prior art in order to patentably distinguish the claimed invention from the prior art.” Applicants have pointed out the various structural differences between the prior art and claims 1-10 and 13-15, namely, the attached executable module, its configuration, and the true “self-destruct” function. Applicants submit that the prior art structure is not “capable of performing the intended use”, creating a self-destructing document, and Ji-MacPhail does not have a true self-destruct capability as required.

D. No *Prima Facie* Case of Obviousness for Claims 1-10 and 13-15

The Examiner has failed to establish a *prima facie* case of obviousness using Ji and MacPhail. Ji purports to disclose a virus detection system that scans the contents of an e-mail system for “detection and elimination of viruses”(Abstract). MacPhail purports to delete expired data by scanning system contents for “managing the retention and deletion of electronic documents” (Abstract). Combining the two references yields a system that scans its contents for viruses and expired data for deletion, as Ji does not enhance or affect the deletion of time-expired data, and MacPhail does not enhance or affect the virus detection and elimination capabilities of Ji. There is no motivation to combine the references as there is no benefit realized.

The Examiner’s mere assertion that it is obvious to combine the references, without finding a suggestion in the prior art, is insufficient to indicate a suggestion or motivation to combine Ji and MacPhail. The suggestion must be express or implied and the Examiner must present a convincing line of reasoning why the references should be combined, which the Examiner has not done. The Examiner states that the suggestion to combine Ji and MacPhail comes from the desire “to create a messaging system with self-destruct module attach to an Email message or embedded to the Email as a design choice”, an argument based on the contentions that Ji taught an “email system with a virus analysis and treatment module wherein the virus (executable code for self-destruction) is attached to Email or other program” and MacPhail taught “delet[ing] an electronic document with an expiration date.” Both contentions, however, are wrong, as discussed above, leaving the Examiner’s argument without merit. There cannot be a suggestion to combine references and create elements that are not present.

The Examiner, in hindsight, is impermissibly finding elements from the specification and putting them into the prior art. The module also is not an executable module configured to instruct a computer to automatically delete a message to which the executable module is attached when the message, based upon a preselected expiration date, is expired, as recited in claim 6.

As for the second requirement of a *prima facie* case of obviousness, the Examiner has not found a reasonable expectation that combining the scanning system of Ji and the scanning system

of MacPhail would produce an executable module that automatically deletes a document or e-mail message, and attaching such a module to a document or e-mail message as recited in claims 1 and 6. It is impossible, and therefore unreasonable, to expect success in combining Ji and MacPhail to create an executable module that automatically deletes a document or e-mail message, and attaching such a module to a document or e-mail message when neither reference teaches or even suggests creating or attaching an executable module.

Regarding the requirement that the prior art teach or suggest all the claim limitations, Applicants have explained how the cited references fail to recite all the limitations in claims 1-10 and 13-15. None of the independent claim limitations have been taught by the prior art, and further, the claim limitations in claim 10 have still not been pointed out in the prior art. Thus, the Examiner has failed to prove a *prima facie* case of obviousness.

IV Rejections Based on the Ji-MacPhail-Shear Combination

A. U.S. Patent No. 5,410,598 (Shear)

Shear purports to disclose a data usage metering, billing, and security system that maintains encrypted data and monitors access by measuring the quantity of decrypted data. The measured quantity is used to calculate a fee for using the data. Access is limited by usage and is prohibited once the amount of usage expires. The system may include a 'self-destruct' feature that disables system operation upon occurrence of a predetermined event and that is implemented by an executable module that is not attached to the data being destroyed.

B. Rejection of Claims 17-47

Claims 17-47 were rejected under 35 U.S.C. §103 as being unpatentable over Ji in view of MacPhail and further in view of U.S. Patent No. 5,410,598 to Shear. The Examiner stated that the Ji-MacPhail combination discloses a self-destructing email messaging system comprising an executable module configured to instruct a computer to automatically delete an email message to which the executable module is attached when a predetermined condition is met; an email messaging system configured to create the message and to transmit the message and attaching the executable module to the message prior to transmission. According to the Examiner, Ji-

MacPhail fail to teach that predetermined condition selected from the group consisting of attempt to print, copy, forward the message, but Shear teaches this well-known technique in the network security art such as a security database system with encryption and decryption data including the self-destruction option when user attempt to access an unauthorized feature, and that it would have been obvious to incorporate the message automatically deleted based upon an attempt to access an unauthorized feature as taught by Shear into Ji-MacPhail system in order to prevent the unauthorized data processing on the network.

In the Response dated August 20, 2001, Applicants argued that, as discussed above, Ji-MacPhail does not disclose a self-destructing e-mail messaging system as recited in claim 17. As for the missing elements, Applicants argued that Shear is not implemented by an executable module attached to the data being destroyed. Applicants further argued that combining Ji-MacPhail with Shear results in an e-mail system that polls its contents and keeps track of data usage, not a self-destructing e-mail messaging system as claimed, and that there is no motivation to combine Ji-MacPhail with Shear, since Ji-MacPhail deals with viruses and expiration dates in e-mail systems (communications), whereas Shear is concerned with the unrelated field of data usage and metering.

With respect to Claims 18-47, Applicants argued that no prima facie case was made at least against claims 18 and 19, 21-23 and 32 because the Examiner has failed to point out where their limitations can be found in the Ji-MacPhail-Shear combination.

With regard to Applicants' arguments against the rejections based on the Ji-MacPhail-Shear combination and claims 17-47, the Examiner states again that "a recitation of the intended use of the claimed invention must result in a structural difference between the claimed invention and the prior art in order to patentably distinguish the claimed invention from the prior art." The Examiner stated that Ji taught an email system with a virus analysis and treatment module wherein the virus (executable code for self-destruction) is attached to Email or other program; MacPhail taught the method of deletion an electronic document with an expiration date; Shear taught the encrypted database and self-destruct feature set by an expiration date; and that it would

have been obvious to combine Ji-MacPhail-Shear to provide the messaging system the options to create a self-destruct document either by an expiration date or by attachment program.

However, Shear does not teach a self-destruct feature, as suggested by the Examiner. Shear purports to disclose a data usage metering, billing and security system that maintains encrypted data and monitors access by measuring the quantity of decrypted data to calculate a usage fee. Access is limited by usage and is prohibited once a predetermined amount of usage expires. The "self-destruct" feature, however, is not implemented by an executable module attached to the data being destroyed. As described at column 18, line 55-68, the "self-destruct" feature automatically destroys a critical part of the system at a preset real time deadline or whenever the customer exceeds a predetermined usage limit unless the customer implements an "antidote" prior to the deadline. This feature resides on and operates remotely from the host computer handling "all database transactions", and the instructions directing the processor to delete information does not come from the information itself. It is the resident software on the host computer providing the interface for all database transactions that instructs the processor to delete data. The information is on medium 100, a separate component from the host computer (see Fig. 1). The software executing the deletion is not even on the same storage media as the information it deletes, so the information cannot be considered to be destroying itself. Therefore, it is simply incorrect to state that Shear taught the self-destruct feature.

Combining Ji and MacPhail creates a system that scans its contents for viruses and expired data for deletion. Shear measures data usage for billing purposes, but not by scanning its contents. Because Shear does not scan, it would not change the scanning functions of Ji-MacPhail. Neither Ji nor MacPhail measure data usage or access and neither system is concerned with billing. Ji-MacPhail, therefore, adds nothing to Shear, and Ji-MacPhail-Shear results in an e-mail system that polls its contents for expired data by usage criteria and disables the expired data. There is no self-destruct function and no attached executable module.

Specifically, claim 17 has patentable limitations that are not taught in the prior art, namely an "executable module configured to instruct a computer to automatically delete an e-

mail to which the executable module is attached when a predetermined condition is met,” and “an e-mail messaging system configured to create the message and to transmit the message”, the executable module being attached to the message prior to transmission. There are also other differences unique to claim 17, namely “an executable module configured to instruct a computer to automatically delete an e-mail message when a predetermined condition is met.” Thus, Claim 17 clearly “result[s] in a structural difference between the claimed invention and the prior art.”

Claims 18, 19 and 43-47 depend from and include the limitations of claim 1 and are therefore allowable for the same reasons presented above with respect to claim 1. Moreover, the Examiner did not point out any grounds for rejection of claims 18 and 19 with respect to the unique limitations recited therein, namely encryption and an executable module configured to perform decryption or deletion, discussed above.

The Examiner again does not acknowledge the limitations in claims 20-43 different from the other claims and rejects both sets of claims on the same basis. There is no indication how or where the prior art combination of Ji-MacPhail-Shear teaches or suggests a virtual container residing in contiguous locations in an electronic storage media of a computer, a header portion or a digital object portion. Perhaps, based on the Examiner’s prior conclusion that the virtual container is the same as the executable module, the Examiner sees the “virus (executable code for self-destruction)” as the virtual container. However, it has already been shown that the virus in Ji is not the executable module of the present invention, and it follows that the virus of Ji cannot be the virtual container of the present invention in claims 20-43.

C. No *Prima Facie* Case of Obviousness for Claims 17-47

There is no motivation to combine Ji-MacPhail with Shear. Ji-MacPhail deals with viruses and expiration dates in e-mail systems (communication), whereas Shear is concerned with data usage and metering (database access), and the two fields are unrelated. As discussed above, combining Ji and MacPhail results in a system that scans its contents for viruses and expired data for deletion. Neither Ji nor MacPhail measure data usage or access or is concerned with billing. Shear measures data usage for billing purposes, but not by scanning its contents.

Shear does not scan, and therefore, would not change the scanning functions of Ji-MacPhail. Combining Ji-MacPhail with Shear would not create any different, special, cumulative, or extra benefit. If two systems produce the same result together as they do separately, then there is no reason to combine them and hence, no suggestion to do so, and the Examiner has provided none.

There can be no reasonable expectation of success that combining Shear with Ji-MacPhail would produce a self-destructing e-mail messaging system as recited in claim 17-19 and 44-47 with an attached executable module, or with a virtual container as in claims 20-43 when the references are structured differently and do not teach or suggest the elements of the claimed invention. Applicants have sufficiently shown that Ji-MacPhail does not provide such a reasonable expectation, and Ji-MacPhail-Shear suffers from the same shortcomings.

The third requirement for a *prima facie* case of obviousness is the teaching or suggestion of all the claim limitations in the prior art. Applicants have already pointed out how the Ji-MacPhail-Shear combination does not recite all the limitations of claims 17-47. Furthermore, the claim limitations in claims 18 and 19, namely encryption and an executable module configured to perform decryption or deletion, still have not been pointed out in the prior art. Similarly, the elements of claims 20-43 have not been specifically shown in Ji, MacPhail or Shear, either alone or in combination

V. Rejections Based on the Hansen-Beck Combination

A. The Hansen Article

Hansen purports to discuss enhancing digital documents with embedded functions, for example, embedding a graphics animation into a digital birthday card and relates to the problems of embedding programs in documents. Hansen, on page 24, second paragraph, describes embedding programs by stating:

It is important to distinguish the notion of enhancing a document with a script from the various authoring languages for educational tools....With the latter, the author constructs a program which *generates* a sequence of images; this contrasts with the enhanced document approach of Ness where the program [sic] within the images. The crucial difference is in user control: *with an enhanced document the reader is in control* and can

employ ordinary text operations to move through the text. [emphasis added]

It is clear from the passage above, “with an enhanced document the reader is in control,” the purpose of Hansen is not enhancing the security of documents, but rather enhancing their aesthetic effect. By putting the reader in control, all security is completely circumvented because the creator of the document has no control over the reader’s actions. Instead, Hansen teaches “Trust” as the “best and only protection.” The system of Hansen does not teach or suggest a security system implemented by any computer functionality.

B. Rejection of Claims 1-10, 13-15 and 17-47

Claims 1-10, 13-15, and 17-47 are rejected under 35 U.S.C. § 103 as being unpatentable over an article by Hansen (the Hansen article) in view of Beck. The Examiner stated that Hansen discloses a method for creating a self-destructing document, comprising the steps of creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based on a preselected expiration date is expired; attaching the executable module to the document, but that Hansen fails to detail when preselected expiration date is expired. The Examiner stated that Beck discloses a Email message with attachment automatically deleted by a time limit and encryption and decryption keys, and it would have been obvious to combine the technique of a email message automatically deleted by an expiration date as taught by Beck and Hansen’s system to improve the security and reliability for message storage and transaction between client/server.

In the Response of August 20, 2001, Applicants argued that Hansen does not disclose a method for creating a self-destructing document as asserted by the Examiner, as Hansen does not mention or even suggest using the embedded functions to implement a method or system for self-destruction of a file in any form. The technology covered in the article is irrelevant and unrelated to the claims of the application. Applicants argued that Hansen does not disclose automatically deleting a document, e-mail message, or any data for that matter, and only alludes to deleting a document as a function to prevent, not as an intended result. There is nothing in Hansen about expiration based on time, a date, or any other criteria.

Applicants also argued that Beck does not “disclose a Email message with attachment automatically deleted by a time limit,” as maintained by the Examiner. Beck clearly states “an attachment may in alternative preferred embodiments be automatically deleted after ... a given time limit, such as 90 days” (Col. 7, ln. 5-9). The attachment is deleted, not the e-mail, which remains untouched. Furthermore, there is no suggestion that the attachment is deleted by an executable module attached to the attachment or the e-mail.

With respect to claims 18-43, Applicants pointed out that the Examiner has not addressed certain distinctive elements of these claims at all, namely the encryption elements of claims 18 and 19, the methods for creating a virtual container, extracting a document from a virtual container, creating a virtual container and extracting a digital object from a virtual container, transmitting a destructible document of claims 20, 21, 23 and 32, respectively, the virtual container system of claim 22. Applicants argued that there is simply nothing in Hansen that even remotely suggests the virtual container embodiment of claims 20-43.

In responding to Applicants’ arguments regarding the Hansen-Beck combination, the Examiner stated again that “a recitation of the intended use of the claimed invention must result in a structural difference between the claimed invention and the prior art in order to patentably distinguish the claimed invention from the prior art.” In this case, according to the Examiner, Hansen taught the multimedia mail wherein the author wants to dynamically create an object or a script which delete a file, Beck taught an email system with attachment, and it would have been obvious to combine Hansen and Beck to provide a messaging system with attachment that could delete or self-destruct as design by author.

Contrary to the Examiner’s assertion, however, Hansen makes no mention of creating a script to delete a file. The only reference to deletion in Hansen is in the context of a function to prevent, rather than an intended result. In fact, the structure of Hansen and its intended use are completely different from that of the present invention. Hansen’s abstract states: “This paper sketches the problems of embedding programs in documents and reviews the solutions adopted in the Ness component of the Andrew ToolKit.” Embedding programs in documents is not the

same as attaching the claimed executable module, since according to Hansen “with an enhanced document the reader is in control.” The purpose of Hansen is not enhancing the security of documents, but rather enhancing their aesthetic effect. By putting the reader in control, all security is completely circumvented because the creator of the document has no control over the reader’s actions. Thus, Hansen unequivocally and specifically places control in the hands of the user. By contrast, in claim 1, control over the document is placed in the attached executable module, which “instructs a computer to automatically delete the document to which the executable module is attached,” not in the user. In the claimed invention, the user has no control over the retention or deletion of the document.

Hansen teaches “Trust” as the “best and only protection,” and the system of Hansen does not teach or suggest a security system implemented by any computer functionality, let alone an executable file. It is completely inaccurate to say that “Hansen taught the multimedia mail wherein the author wants to dynamically create an object or script which delete a file.” Hansen never teaches the author of a script to delete a file. Hansen plainly states on p. 28 that “Embedding ... scripts ... does not introduce a new ... security problem, but makes obvious a common security problem. The problem is that...when I execute...[the script]...it may do anything...in particular delete a file...or send a copy of a...forth-coming examination – to...a student about to take that examination. Since a Ness script is a program...*its execution is a security loophole.*” (emphasis added) Yet, the Examiner maintains that the execution of a script provides enhanced security, like the present invention.

Not only does Hansen not teach any method of using a script to enhance the security of a document, the above passage is intended to help and protect the receiver of the document containing the script. The warning states “**The author of the script or program – if malicious – can write it in such a way that *it can destroy your files.* If you do not trust the place or person from which you got his script, *DO NOT EMPOWER IT.***” (emphasis added) and is directed to the receiver, advising him to not execute the script if he does not trust the source. Accordingly, *deleting or destroying data is never taught as an intended result in Hansen.*

The method in claim 1 requires “creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based upon a preselected expiration date, is expired” and “attaching the executable module to the document.” The executable module is attached to the document, making it part of the document, and is not functional without the document to which it is attached. The document may function without the attached executable, but a document by itself is not the present invention. The executable module “instructs a computer to automatically delete the document to which the executable module is attached.” The instructions that make up the executable module, now part of the overall document, direct a processor to delete the document. Since the executable module is a part of the document, the instructions to delete the document come from the document itself, thereby accomplishing a true “self-destruct,” which is the intended use of the present invention.

In Hansen, there is no automatic deletion, no executable module that deletes anything, no expiration date, no expired data, and no attaching of an executable module that deletes its attached document. Applicants submit that Hansen, which discusses embedding scripts into documents to enhance their appearance and ornamental effects, not attaching executable modules to documents to perform self-destruction, has been misinterpreted by the Examiner. Security is discussed as a concern in regard to the damage that the script can do to a system, not as a concern in regard to the security of a document, not even in regard to the security of the document the script is embedded in. Therefore, Hansen does not teach “the multimedia mail wherein the author wants to dynamically create an object or a script which delete a file.”

There is no “attachment which could delete or self-destruct as design by author.” Beck does not have a self-destructing attachment and neither does Hansen. Applicants have shown that Hansen does not even teach deletion or destruction of files, much less self-destruction.

Applicants have clearly pointed out how “a recitation of the intended use of the claimed invention” does, in fact, “result in a structural difference between the claimed invention and the prior art,” thereby “patentably distinguish[ing] the claimed invention from the prior art.”

Accordingly, Hansen cannot perform the intended use of creating a self-destructing document as recited in claim 1. Accordingly, claim 1 clearly “result[s] in a structural difference between the claimed invention and the prior art...”

Claims 2-5, 18, and 44-47 depend from and include all the limitations of claim 1 and are thus allowable. Applicants point out that the Examiner’s Response to Arguments still do not point out any grounds for rejection of claim 18 with respect to their unique limitations.

Claim 6 recites “an e-mail messaging system comprising ... an executable module configured to instruct a computer to delete a message to which the executable module is attached.” The executable module, as claimed is configured to specifically delete a message it is attached to. There is no such deleting script, or executable file, in Hansen. The structural differences of claim 1 are shared by claim 6, and accordingly, claim 6 is structurally and patentably distinguished from Hansen-Beck and therefore, allowable.

Claim 6 also “result[s] in a structural difference between the claimed invention and the prior art.” Claims 7-10, 13-15 and 19 depend from and include all the limitations of claim 6 and are therefore allowable for the same reasons as claim 6. Applicants note that the Examiner’s Response to Arguments still do not point out any grounds for rejection of claims 10 and 19 with respect to the unique limitations recited therein.

Claim 17 also recites a self-destructing e-mail messaging system with similar limitations to claim 6, except that the executable module of claim 17 is “configured to instruct a computer to automatically delete an e-mail message to which the executable module is attached when a predetermined condition is met, wherein said predetermined condition is selected from the group consisting of an attempt to print the message, an attempt to copy the message and an attempt to forward the message.” Claim 17, therefore, has the same structural differences from Hansen-Beck as claim 6 and the cited references cannot perform the intended use of claim 17.

Claims 20-43, the unique limitations of which the Examiner fails to address, recite a

virtual container with a header portion and a digital object portion residing in contiguous locations in an electronic storage media of a computer. Hansen has nothing even remotely similar. Scripts are embedded into different points of documents in Hansen. Assuming, for arguments sake, that the document of Hansen is a digital object, there still is no similarity in structure. Where claims 20-43 insert the document (digital object) into a container, Hansen inserts scripts into the document. The document in Hansen is not inserted into anything, and it is not contained by anything. The document does the exact opposite and receives, or contains, scripts. The structure, therefore, of Hansen and claims 20-43 are opposites.

C. No *Prima Facie* Case of Obviousness for Claims 1-10, 13-15 and 17-47

The Examiner has not made a prima facie case of obviousness against the pending claims using Hansen and Beck. There is no suggestion to combine Hansen and Beck in the prior art, and the Examiner finds reason to combine them by stating "it would have been obvious ... to combine Hansen and Beck to provide a messaging system with attachment which could delete or self-destruct as design by author."

However, Hansen and Beck are not at all related. Hansen discusses embedding scripts to enhance documents with a specific scripting language, Ness, and Beck deals with the problem of limited bandwidth and storage capacity on e-mail networks by attaching a network address to an e-mail. One skilled in the art who wanted to create documents with added ornamental function would look to Hansen and not be concerned with the bandwidth and storage capacities of a network that would not help in writing a script and embedding it into a document. By the same token, one skilled in the art who desired a more efficient e-mail attachment transmission method would not write scripts into the documents to reduce congestion on network lines, as that would only add more traffic, since embedding scripts in a document adds more data to the document, thereby increasing its size, and larger documents take up more bandwidth for transmission and more storage capacity by occupying more memory. The two references are counter-intuitive and defeat each other's objectives and goals. Accordingly, there is no logical reason to combine the two references when their true nature and disclosure is understood.

There also is not a reasonable expectation of success that combining the two references would create an executable module that automatically deletes a document or e-mail message to render claim 1 obvious and attaching such a module to a document or e-mail message to render claim 6 obvious. Applicants cannot understand how two references that do not teach creating an executable file, attaching the executable file, or automatically deleting an expired document can have any chance of producing a self-destructing e-mail with those very elements.

That there is no reasonable expectation of success for producing the invention of claims 20-43 is even more certain because those claims teach the opposite of the Hansen structure where documents contain scripts. Claims 20-43 teach a virtual container that contains a digital object which may be a document. The “document” of claims 20-43 is the containee, not the container. Hansen is opposite, as the enhanced document is the container of embedded scripts. Beck is irrelevant to claims 20-43 anyway since those claims do not recite attachments. Therefore, because Hansen and claims 20-43 actually recite opposite structures, there is no chance of success in producing the virtual container embodiment of the present invention, claimed in claims 20-43, from combining Hansen and Beck.

Finally, there is no teaching or suggestion of all the claim limitations in the prior art. Applicants’ arguments above address the many deficiencies in the cited references for reciting all the limitations in claims 1-10, 13-15, and 17-47. None of the independent claim limitations have been taught by the prior art. Further, the Examiner still ignores the claim limitations in claims 10, 18, 19 and 20-43, as there is no indication or discussion in any of the Examiner’s communications of the elements of these claims and where they can be found in the prior art.

VI. Rejections Based on the Drake-Norin Combination

A. U.S. Patent No. 6,006,328 (Drake)

Drake purports to disclose a security system for computer software, wherein certain attacks on executable software are prevented by replacing executables with new versions having enhanced security. The new executables are not attached to anything. In addition, the only deletion function in Drake operates by intentionally storing specific information in a particular

area of memory whose contents are deleted upon an unexpected use of memory, thereby deleting the specific information. There is no destruction caused “to the system both hardware, software or the embedded program itself.” There is no embedded program, and nothing destroys itself.

B. U.S. Patent No. 5,787,247 (Norin et al.)

Norin involves a data replication system where a number of copies of certain data are maintained simultaneously on network servers. Norin purports to disclose a system for ensuring that changed data is not lost by being inadvertently deleted without storing the changes somewhere else by verifying that changes made to a local copy of the data reside on at least one other system in the network.

At column 4, lines 14 to 44, Norin explains a typical prior art replication process where each server keeps track of locally made changes to a data set and broadcasts those changes to other servers to update their copies. If a change occurs in a data set and it is deleted for some reason before the changes are broadcast, the changes will be lost. Norin uses server communication to keep track of changes in data and their location, and its system is concerned with keeping track of data that is changed or deleted, not the actual deletion itself.

C. Rejection of Claims 1-10, 13-15 and 17-47

Claims 1-10, 13-15, and 17-47 have been rejected under 35 U.S.C. § 103 as being unpatentable over U.S. Patent No. 6,006,328 to Drake in view of U.S. Patent No. 5,787,247 to Norin et al. The Examiner states that Drake discloses a method for creating a self-destructing document, comprising the steps of creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based on a preselected expiration date is expired; attaching the executable module to the document (such as a message with a header is attached by a executable code or software which is designed to self-destruct). However, the Examiner states that Drake fails to detail the a preselected expiration date is expired, and Norin discloses a Email message with time-based expiration date wherein an object is older a set time will be deleted automatically. Therefore, according to the Examiner, it would have been obvious to combine the technique of a email

message automatically delete by an expiration date as taught by Norin and Drake's system to would improve the reliability for data storage and transaction between client/server.

In the August 20, 2001 response, Applicants argued that Drake does not disclose a method for creating a self-destructing document as alleged by the Examiner but rather discusses a security system for computer software to prevent certain attacks on executable software by persons or other software. There is no attached executable in the entire reference.

The Examiner based his rejection of claims 20-47 on the "same rationale applied above" for rejecting claims 1-5. The Applicants argued that, Claims 20-47 do not share any similarities with claims 1-5, and the Examiner has not indicated where the elements of claims 20-47 can be found in either reference singularly, or in combination, since neither reference even suggests the method for creating a virtual container of claim 20, the method for extracting a document from a virtual container of claim 21, the virtual container system of claim 22, the method for creating a virtual container and extracting a digital object from a virtual container of claim 23, or the method for transmitting a destructible digital object of claim 32.

In response to Applicants' arguments against the rejections based on the Drake-Norin combination, the Examiner states, once again, that "a recitation of the intended use of the claimed invention must result in a structural difference between the claimed invention and the prior art in order to patentably distinguish the claimed invention from the prior art." The Examiner maintained that Drake discloses a method for detecting or preventing Virus, Worm, macro recorders or hacker attack a computer system by hidden or embed or attach the program which cause the destruction to the system both hardware, software or the embedded program itself; Norin taught the data object or message is deleted automatically by a set time; and it would have been obvious to combine Drake and Norin to provide a messaging system with an automatically deleted or self-destructing feature that attaches or embeds to an message.

Drake, however, does not disclose "a method for detecting or preventing Virus, Worm, macro recorders or hacker attack a computer system by hidden or embed or attach the

code/program which cause the destruction to the system both hardware, software or the embedded program itself”, as alleged by the Examiner. The Examiner appears to find embedded or hidden code/program that causes destruction to hardware, software or the code/program itself. Applicants, however, are at a loss to find such a code/program in the Drake patent. Code is not hidden, embedded or attached in Drake. Drake merely places a buffer (secure drivers 51) between the user and the software to be protected so that “the user 23 no longer communicates with the operating system 17 or the unprotected computer hardware 18.” As a result, “the rogue program 41 can no longer eavesdrop on ID-Data.”

The first method of Drake that places a secure buffer between user and software does not hide, embed or attach the secure buffer. Column 9, line 53 of Drake states “user 23 now communicates directly with secure drivers 51 ... part of the secure ID-Data entry program code 31 ... utilised [sic] by the security enhance (eg. tamper protect) application code 52.” Adding security features and enhancing pre-existing application files is not embedding, hiding or attaching application files. The secure drivers are not embedded, hidden or attached. The secure drivers are part of the secure ID-Data entry code used by the security enhanced application.

The second method used by Drake that places a secure buffer between user and software modifies and replaces non-secure executable files. In this method, described at column 14, lines 6-26, an applicator program accepts and modifies the old executable and replaces it with a new executable having encryption and safe equivalent code, among other security enhancements. This is not hiding, embedding, or attaching the executable file, it is merely replacing the executable file. Accordingly, Drake does not prevent software attacks “by hidden or embed or attach the code/program,” as alleged by the Examiner.

With regard to the Examiner’s statement that Drake teaches “the code/program which cause the destruction to the system both hardware, software or the embedded program itself”, Drake at col. 7, lines 43-52 purports to disclose a mechanism that destroys software by storing the software in a specific part of memory that is erased when a software attack is detected. In effect, Drake stores software on memory that is deleted by the debugger so that when the

debugger is activated to do a trace, it will destroy the software. However, the software does not destroy itself, but rather the debugger does. There is no “code/program which cause the destruction to the system both hardware, software or the embedded program itself” in Drake. The code/program that destroys data in Drake is the debugger, which is not an element of Drake, but of a rogue program trying to sabotage protected software. There is no “hidden or embed or attach ... code/program which cause the destruction to ... the embedded program itself” as well. Drake does not provide a self-destruct function.

Norin involves a data replication system where a number of copies of certain data are maintained simultaneously on network servers. Norin uses server communication to keep track of changes in data and their location. Norin never discusses deleting data directly by its own system, which is concerned with keeping track of data that is changed or deleted, not the actual deletion itself. The Examiner alleged that “Norin taught the data object or message is deleted automatically by a set time.” Norin discusses deleting time-expired data at Column 24, lines 1-51, where the processing block does not delete time-expired data. The act of deleting time-expired data is discussed as it would impact the replication process, as stated, “special consideration must be given to the handshaking process,” meaning the handshaking process between the replication process and the deletion mechanism.

Norin poses the question “as to whether a replica node that once held changes but has now expired and deleted them should respond to a replica delete pending packet requesting verification that the changes which have expired exist in the network.” The sole purpose of discussing deletion of time-expired data is to examine its impact on a replication process. Therefore, since Norin does not directly teach the deletion of time-expired data and how it should be accomplished, it seems that the scope of Norin’s contribution to the concept is no more than the general prior art discussed earlier, which has been distinguished.

Claim 1 requires “creating an executable module which instructs a computer to automatically delete the document to which the executable module is attached when the document, based upon a preselected expiration date, is expired” and “attaching the executable

module to the document.” The executable module is attached to the document, making it part of the document, and is not functional without the document to which it is attached. The document may function without the attached executable, but a document by itself is not claimed. The executable module “instructs a computer to automatically delete the document to which the executable module is attached.” The instructions that make up the executable module, now part of the overall document, direct a processor to delete the document. Since the executable module is a part of the document, the instructions to delete the document come from the document itself, thereby accomplishing a “self-destruct”.

By contrast, in Drake, the only mention of deleting data is concerned with tracing a rogue attack and executed by a debugger, not by the data being deleted. There is no “self-destruct” and no instructions originating from the data itself directing a processor to execute a deletion process. Therefore, Drake does not teach “hidden or embed or attach the code/program which cause the destruction to the system both hardware, software or the embedded program itself” as maintained by the Examiner. Accordingly, Drake cannot perform the intended use of creating a self-destructing document or e-mail message as claimed. Claim 1 clearly “result[s] in a structural difference between the claimed invention and the prior art...”

Claims 2-5, 18, and 44-47 depend from and include all the limitations of claim 1 and are therefore also patentable. Applicants note that the Examiner has not pointed out any grounds for rejection of claim 18 with respect to the unique limitations recited therein, discussed above.

Claim 6 recites “an e-mail messaging system comprising ... an executable module configured to instruct a computer to delete a message to which the executable module is attached.” The executable module, as claimed, is configured to specifically delete a message it is attached to. Executables are not created or configured for individual documents in Drake or Norin. Drake creates executables for executables, to replace them. The Examiner contends that “Norin taught the data object or message is deleted automatically by a set time.” Norin, however, does not teach automatically deleting a data object by a set time. The reference only touches upon a prior art deletion mechanism and how it should be handled by the Norin replication

process. Norin never explains a deletion mechanism or how one would work.

The system of claim 6 also comprises an “e-mail messaging system configured to create an e-mail message and to transmit the e-mail message, the e-mail message system attaching the executable module to the e-mail message prior to transmission.” The Examiner relies on Drake to “attach the code/program.” Applicants have explained above that Drake does not attach code or program to anything. Thus, Claim 6 “result[s] in a structural difference between the claimed invention and the prior art...”

Claims 7-10, 13-15 and 19 depend from and include all the limitations of claim 6 and are therefore allowable for the same reasons as claim 6. Applicants note that the Examiner still does not point out any grounds for rejection of claim s 10 and 19 with respect to the unique limitations recited therein.

It is clear, from the above, that the “recitation of the intended use of the claimed invention” in claims 1-10, 13-15 and 18-19 does, in fact, “result in a structural difference between the claimed invention and the prior art.” The Drake structure is not capable of creating a self-destructing document, e-mail message, or any type of self-destructing file. Norin does not help any because, contrary to the Examiner’s position, it does not contribute any teaching of a deletion mechanism.

Claim 17 also recites a self-destructing e-mail messaging system with limitations similar to those of claim 6 except that the executable module of claim 17 is “configured to instruct a computer to automatically delete an e-mail message to which the executable module is attached when a predetermined condition is met, wherein said predetermined condition is selected from the group consisting of an attempt to print the message, an attempt to copy the message and an attempt to forward the message.” Claim 17, therefore, has the same structural differences from Drake-Norin as claim 6, and the cited references cannot perform the intended use of claim 17.

The Examiner also states “If the prior art structure is capable of performing the intended

use, then it meets the claim.” Applicants submit that the prior art structure is not capable of performing the intended use, creating a self-destructing document. It has been shown that Drake does not have a true self-destruct capability as claimed.

With regard to claims 20-43, the Examiner has not provided a separate grounds of rejection. However, as discussed above, claims 20-43 are very different from claims 1-19. The virtual container embodiment recited in claims 20-43 actually contains the e-mail or document. The Examiner still does not acknowledge the different limitations of claims 20-43 from the other claims and rejects both sets of claims on the same basis. Among the unique limitations in the virtual container embodiment are “contiguous locations,” “a header portion,” and “a digital object portion”. Claim 20 recites “a header portion” and “a digital object portion”. Claim 21 recites a method for extracting a document from a virtual container. Claim 22 recites a virtual container system to implement the method steps in claims 20 and 21. Claim 23 recites a method for creating a virtual container and extracting a digital object from a virtual container, combining claims 20 and 21. Claim 32 recites a method for transmitting a destructible document. Claims 21-23 and 32 all recite the header portion, digital object portion, and the contiguous locations in an electronic storage media of a computer as in claim 20. Accordingly, they all share the same structural differences from the prior art, and the prior art combination of Drake-Norin does not teach or suggest a virtual container residing in contiguous locations in an electronic storage media of a computer, a header portion, or a digital object portion.

D. No *Prima Facie* Case of Obviousness for Claims 1-10, 13-15 and 17-47

There is no suggestion to combine Drake and Norin in those references or anywhere in the general prior art. The Examiner maintains “it would have been obvious ... to combine Drake and Norin teaching to provide a messaging system with an automatically deleted or -self-destructing feature which attach or embed to an message.” The Examiner bases this conclusion on his interpretation of Drake as teaching “a method for detecting or preventing Virus, Worm, macro recorders or hacker attack a computer system by hidden or embed or attach the code/program which cause the destruction to the system both hardware, software or the embedded program itself.”

The Examiner's logic is mistaken, since the very purpose of Norin is counter-intuitive to the objectives of the present invention. Norin prevents inadvertent deletion, whereas the present invention promotes deletion, deletion of a document or e-mail message that is expired.

Even in the general prior art, there is no suggestion to combine Drake and Norin. Drake is concerned with protecting the integrity of software system files, protecting software systems from hacker attacks. Norin is concerned with data loss, not by an intentional attack, but by mistake in a data replication system. The main purpose of Norin is not providing a safeguard against attack, but a safeguard against data loss by ensuring that copies of changed data exist before deletion, thereby preventing a loss of changed data. Accordingly, Norin would serve as a reference to an artisan looking to create a replica administration system that ensures copies of recently changed data exist before deleting certain data so as not to lose any changes made. Norin discusses the type of system in which it would be employed at col. 4, ln. 12, "The preferred replication process...is an asynchronous store and forward replication process." Because Drake does not offer any help or improvement in ensuring data changes are preserved and replicated, the artisan trying to create a replica data administration system would not look to Drake. If he did, he would find nothing helpful. Drake enhances security from rogue attacks by replacing pre-existing executables and placing secure communication facilitators between potentially dangerous users and protected software.

The reverse is true as well. An artisan looking to improve the security of his particular system against rogue, hacker attacks would look to Drake and possibly take the methods disclosed therein for enhancing security, namely, replacing pre-existing executables and placing secure communication facilitators between potentially dangerous users and protected software.

The second requirement in establishing a *prima facie* case of obviousness against Drake-Norin calls for a reasonable expectation of success that combining the security system of Drake with the data replication system of Norin would produce the claimed invention. However, Drake purports to disclose a secure communicating buffer and security-enhanced executables. The secure communicating buffer is irrelevant to the claimed invention and thus can be disregarded.

Only the security-enhanced executable replacements are left. Norin purports to disclose a replication system that uses server communication to keep track of changed data and its location to prevent inadvertent data loss by deletion. The reasonable expectation, therefore, should be that a security-enhanced executable replacement combined with a server communication network for keeping track of changed data with the aim of preventing inadvertent deletion, produces an executable module that automatically deletes a document or e-mail message, and attaching such a module to a document or e-mail message as recited in claims 1 and 6.

However, the executable module that automatically deletes a document or e-mail message is not a replacement executable, since it does not replace anything. Replacing executable files with enhanced versions does not hint at attaching executable files to anything. Keeping track of changed data with the aim of preventing inadvertent deletion also does not suggest creating an executable module that automatically deletes a document or e-mail message, or attaching such a module to a document or e-mail message. It seems obvious that keeping track of changed data with the aim of preventing inadvertent deletion is counter-intuitive to an executable module that automatically deletes a document or e-mail message, and attaching such a module to a document or e-mail message. Such an executable module would delete the document or e-mail message regardless of its changes or location. Accordingly, there is no reasonable expectation that combining Drake and Norin will produce the present invention as recited in claims 1-10, 13-15, 17-19 and 44-47.

There is even less of a chance that combining Drake and Norin would produce the virtual container embodiment of claims 20-43. There is nothing similar, in their entire disclosure, to the virtual container residing in contiguous memory locations with a digital object portion and a header portion. Without the virtual container, it is impossible for the cited combination to have any of the other limitations in that set of claims.

The third requirement for a *prima facie* case of obviousness is the teaching or suggestion of all the claim limitations in the prior art. As Applicants' discussions above show, none of the limitations of claims 1-10, 13-15, 17-19 and 44-47 have been taught by the prior art. Moreover,

the Examiner has ignored the limitations in claims 10, 18, 19 and 20-43.

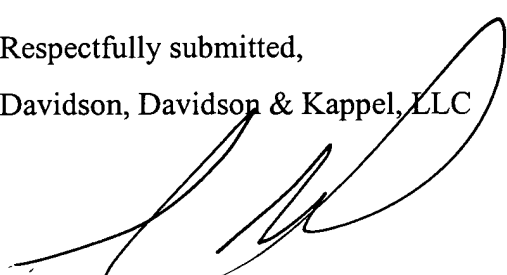
CONCLUSION

The present invention is new, useful, and unobvious. Reconsideration and allowance of the present application is therefore respectfully requested.

Respectfully submitted,

Davidson, Davidson & Kappel, LLC

By :



Cary S. Kappel (Reg. 36,561)
485 Seventh Avenue, 14th Floor
New York, NY 10018
212-736-1940 (phone)
212-736-2427 (fax)